MKDIR

Vulnerable to TOCTOU issues

Sean Barnum, Cigital, Inc. [vita[1]]

2007-03-29

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 9543 bytes

Attack Category	• Path spoofing or confusion problem
Vulnerability Category	• Indeterminate File/Path • TOCTOU - Time of Check, Time of Use
Software Context	• File Creation • File I/O
Location	
Description	The mkdir() function attempts to create a new empty directory. It is generally vulnerable to classic TOCTOU attacks. A call to mkdir() should be flagged if the first argument (the directory) is used earlier in a check-category call.
APIs	**Function Name** / **Comments** _mkdir / use; win32 _tmkdir / use; win32 _wmkdir / use; win32 mkdir / use mkdirp / use; Solaris
Method of Attack	The key issue with respect to TOCTOU vulnerabilities is that programs make assumptions about atomicity of actions. It is assumed that checking the state or identity of a targeted resource followed by an action on that resource is all one action. In reality, there is a period of time between the check and the use that allows either an attacker to intentionally or another interleaved process or thread to unintentionally change the state of the targeted resource and yield unexpected and undesired results. The mkdir() call is a use-category call, which when preceded by a check-category call can be indicative of a TOCTOU vulnerability.

1. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/35-BSI.html (Barnum, Sean)

	A TOCTOU attack in regards to mkdir() can occur when a check for the existence of a directory occurs and then a directory is created. Between those two actions, an attacker could, for example, use a link to link the directory to be created to a known directory. The subsequent creation of the directory would either fail or have unexpected results or behavior.
Exception Criteria	
Solutions	

Solution Applicability	Solution Description	Solution Efficacy
Generally applicable to any mkdir().	Utilize a file descriptor version of stat/ fstat when checking.	
Generally applicable.	The most basic advice for TOCTOU vulnerabilities is to not perform a check before the use. This does not resolve the underlying issue of the execution of a function on a resource whose state and identity cannot be assured, but it does help to limit the false sense of security given by the check.	Does not resolve the underlying vulnerability but limits the false sense of security given by the check.
Generally applicable.	Limit the interleaving of operations on files from multiple processes.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
Generally applicable.	Limit the spread of time (cycles) between the check and use of a resource.	Does not eliminate the underlying vulnerability but can help

			make it more difficult to exploit.
	Generally applicable.	Recheck the resource after the use call to verify that the action was taken appropriately.	Effective in some cases.

Signature Details

int mkdir(const char *pathname, mode_t mode);
int _mkdir(const char *pathname, mode_t mode);
int mkdirp(const char *pathname, mode_t mode);
int _tmkdir(const char *pathname, mode_t mode);
int wmkdir(const wchar_t *pathname, mode_t mode);

Examples of Incorrect Code

```
char theDirectory []="targeteddirectory";
strcpy(theDirectory,"wrongnamedirectory");
mkdir(theDirectory);
```

In this case a directory was created where the name is not the intended one.

It is also possible for a race condition to be created, with unpredictable results.

```
#include <sys/types.h>
#include <sys/stat.h>

int check_status;
int use_status;
struct stat statbuf;
...

check_status=stat("tobecreateddir",
&statbuf);

...
<long enough intervening code>

use_status=mkdir("tobecreateddir",..);
```

Examples of Corrected Code

```
One solution is to eliminate the
pre-deletion test and instead use
a post-deletion status check.

#include <sys/types.h>
#include <sys/stat.h>

int status;
...
status = mkdir("/home/cnd/mod1",
S_IRWXU | S_IRWXG | S_IROTH |
S_IXOTH);
...
```

Source References	• Viega, John & McGraw, Gary. *Building Secure Software: How to Avoid Security Problems the Right Way*. Boston, MA: Addison-Wesley Professional, 2001, ISBN: 020172152X, ch. 9. • man page for mkdir() • Microsoft Developer Network Library (MSDN). • http://www-cse.ucsd.edu/classes/sp05/cse127/Lec10.pdf • http://www.die.net/doc/linux/man/man3/mkdir.3.html
Recommended Resource	
Discriminant Set	**Operating Systems**: • UNIX • Windows **Languages**: • C • C++

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. mailto:copyright@cigital.com